


```
CCCCCCCC 000000 88888888 DDDDDDDD EEEEEEEEE FFFFFFFF
CCCCCCCC 000000 88888888 DDDDDDDD EEEEEEEEE FFFFFFFF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88888888 DD      DD EEEEEEEE FFFFFFFF
CC        00    00 88888888 DD      DD EEEEEEEE FFFFFFFF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88      88 DD      DD EE      FF
CC        00    00 88      88 DD      DD EE      FF
CCCCCCCC 000000 88888888 DDDDDDDD EEEEEEEEE FFFFFFFF
CCCCCCCC 000000 88888888 DDDDDDDD EEEEEEEEE FFFFFFFF
```

....
....
....
....

```
RRRRRRRR EEEEEEEEE QQQQQQ
RRRRRRRR EEEEEEEEE QQQQQQ
RR      RR EE      QQ      QQ
RR      RR EE      QQ      QQ
RR      RR EE      QQ      QQ
RR      RR EE      QQ      QQ
RRRRRRRR EEEEEEEE  QQ      QQ
RRRRRRRR EEEEEEEE  QQ      QQ
RR      RR EE      QQ      QQ
RR      RR EE      QQ      QQ
RR      RR EE      QQ      QQ
RR      RR EE      QQ      QQ
RR      RR EEEEEEEEE QQQQ      QQ
RR      RR EEEEEEEEE QQQQ      QQ
```

REQUIRE FILE: LIB\$;COBDEF.REQ

Version 1-017 BH1017 12-AUG-81

* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
* ALL RIGHTS RESERVED. *

* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
* TRANSFERRED. *

* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
* CORPORATION. *

* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *

DIGITAL.

MODIFICATION HISTORY:

- 1-001 - Start of revision history. RKR 13-SEPT-79
- 1-002 - Add COBSK_EXC_ORG. MWS 4-OCT-79
- 1-003 - Add Minimums and Maximums to series of COBSK symbols.
Redefine module name data structure by making first
word a length of entry.
Add COBSK_EXC_CLRSS (Close Reel)
Add COBSK_ACC_SIZE (Buffer size in COBSACCEPT)
Add COBS_USE_EXIT (Special signal between handler and
COB\$\$INVOKE_USE)
RKR 18-OCT-79
- 1-004 - Add COBSA_NAM_FILES to module name data structure.
RKR 18-OCT-79
- 1-005 - Remove COBS_USE_EXIT. (It needs to be in COBMSG.MDL
so that S.MLB knows about it, so that the .MAR world can
get at it. RKR 20-OCT-79
- 1-006 - Add COBSK_DIS_SIZE. RKR 21-OCT-79
- 1-007 - Change comment on COBSK_ERR_BADCL. RKR 29-OCT-79
- 1-008 - Changed macro COBSL_USE_COUNT into COBSB_USE_COUNT
and added COBSB_GUSE_COUNT which is a count of the global
file entries. Also added COBSA_CHK_PROC which is the
definition of the offset of USE procedure address in the
signal arguments. Note that these changes were made by the
COBOL group during the time when there was only part-time
RTL support provided. Also added comments to the definition

- List for COBSIOEXCEPTION. LB 04-MAR-81
- 1-009 - Added macro definitions for the Data Base USE procedure entries. Also added associated literals for data base USE entries. LB 11-MAR-81
 - 1-010 - Added more field references for data base USE entries. LB 12-MAR-81
 - 1-011 - Changed macro for COBSA_CHK_PROC to be an offset of 12 in the signal array to account for the addition of an FAO count parameter which will reside at offset 8 in the signal array. Same change occurred for the corresponding data base macro (COBSA_DBCHK_PROC). LB 16-APR-81
 - 1-012 - Added a macro COBSA_OPN_PROC to be an offset of 16 in the signal array due to the addition of another parameter being signalled in module COBSIOEXCEPTION. Changed the name of macro COBSA_CHK_PROC to COBSA_FIL_PROC to more accurately reflect the information we are referencing. LB 21-APR-81
 - 1-013 - Added a literal entry to the COBSERROR set of entries (COBSK_ERR_DBARG). Also incremented the maximum value for the case stmt by one to account for the added entry. LB 1-MAY-81
 - 1-014 - Added COBSK_EXC_KEY. Also incremented the maximum value for the case stmt by one to account for the added entry. PDG 24-JUL-81
 - 1-015 - Added COBSK_EXC_UNLCK as new OPEN error for explicit record locking. Also incremented the maximum value for the case stmt by one to account for the added entry. LB 8-AUG-81
 - 1-016 - Fixed spelling error on OPEN error. LB 12-AUG-81
 - 1-017 - Added COBSK_EXC_PIO. Previous IO NOT sucessful READ for DELETE and START statements. Updated COBSK_EXC_MAXM to 7. BH 1-SEP-83

```

!+
! Definitions for the module name data structure.  This structure is used by
! routine COB$FIND_NAME, which is called by COB$CALL and COB$CANCEL.  It is
! built in PSECT COB$NAMES_____2 by a contribution from each COBOL module.
! The bracketing PSECTS COB$NAMES_____1 and COB$NAMES_____3 are used to find
! the beginning and end of the structure.

```

MACRO

COBSL_NAM_LEN= 0,0,32,0 %.	! Length in bytes of name entry
COBSA_NAM_NAME= 4,0,32,0 %.	! Pointer to ASCII module name
COBSA_NAM_ENTRY=8,0,32,0 %.	! Address of module entry point
COBSL_NAM_LLEN= 12,0,32,0 %.	! Length of local storage for module
COBSA_NAM_LOCAL=16,0,32,0 %.	! Address of local storage for module
COBSA_NAM_FILES=20,0,32,0 %.	! Address of counted list of RAB addresses

LITERAL

```
COBSS_NAM=      24;      ! Length of block
                      ! *** Should never be used to
                      ! search this table ****
```

!+
Definitions for the first parameter of COB\$ERROR, the routine called by
compiled code to report an error. If the error is COB\$K ERR SORT, a second
parameter is also present, which is the status returned by SORT.

LITERAL

COB\$K_ERR_MIN=0,	! Minimum for CASEing
COB\$K_ERR_ALTER=0,	! GO TO with no preceding ALTER
COB\$K_ERR_PERFR=1,	! Recursive activation of PERFORM
COB\$K_ERR_PERFN=2,	! Nesting error for PERFORM
COB\$K_ERR_PERFT=3,	! TIMES value overflows longword
COB\$K_ERR_SUBSD=4,	! OCCURS DEPENDING value overflows longword
COB\$K_ERR_SUBSC=5,	! Subscript overflows longword
COB\$K_ERR_SORT=6,	! Error during SORT
COB\$K_ERR_INSPE=7,	! INSPECT CONVERTING lengths unequal
COB\$K_ERR_BADCL=8,	! CALL failed on routine (!AS)
COB\$K_ERR_DBARG=9,	! Expression value in DB arg list overflows longword
COB\$K_ERR_MAX=9;	! Maximum for CASEing

!+
 Definitions for the COBOL file context area.
 The fields COBSA_CTX_LINAG and following are present only if it is
 a lineage file. This area is allocated in PSECT \$LOCAL by the
 compiler for each file immediately preceding the RAB.

MACRO

COBSB_CTX_FLAGS=	-4,0,8,0 %	! Flag bits
COBSV_CTX_OPENL=	-4,0,1,0 %	! TRUE only if a LINAGE file has been opened but not yet written to.
COBSV_CTX_OPT=	-4,1,1,0 %	! TRUE only if it is an OPTIONAL file.
COBSV_CTX_FLK=	-4,2,1,0 %	! TRUE only if File closed WITH LOCK.
!a9a9 COBSV_CTX_PIO=	-4,3,1,0 %	! TRUE only if previous IO was successful READ
!a9a9 COBSB_CTX_FLAG2=	-3,0,8,0 %	! More flag bits
COBSV_CTX_OFNF=	-3,0,1,0 %	! TRUE only if Optional file not found.
COBSV_CTX_NNVR=	-3,1,1,0 %	! TRUE only if No next valid record.
COBSB_CTX_MODE=	-2,0,8,0 %	! OPEN mode
COBSB_CTX_ID=	-1,0,8,0 %	! Version number
COBSL_CTX_LINAG=	-8,0,32,0 %	! LINAGE storage
COBSL_CTX_FOOTI=	-12,0,32,0 %	! FOOTING storage
COBSL_CTX_TOP=	-16,0,32,0 %	! TOP storage
COBSL_CTX_BOTTO=	-20,0,32,0 %	! BOTTOM storage
COBSL_CTX_COUNT=	-24,0,32,0 %	! LINAGE-COUNTER itself

!+
 Values of COBSB_CTX_MODE.

LITERAL

COBSK_CTX_INPUT=0,	! INPUT mode
COBSK_CTX_OUTPU=1,	! OUTPUT mode
COBSK_CTX_I_O= 2,	! I-O mode
COBSK_CTX_EXTEN=3,	! EXTEND mode
COBSK_CTX_MAX=3;	! Maximum for upper bound check

LITERAL

COBSS_CTX_NOLIN=4,	! Size of CTX area without LINAGE
COBSS_CTX_LINAG=24;	! Size of CTX area with LINAGE

!+ Definition for the COBOL specific stack location (relative to FP, mapped as
a REF BLOCK[BYTE]). This location is to be presumed initialized if and
only if SFSA_HANDLER contains COBSHANDLER.

MACRO

COBSA_SF_USE= -4,0,32,0 %; ! Address of USE list

!+ Definition for the USE description list pointed to by COB\$A_SF_USE. This structure is allocated by the compiler in PSECT \$PDATA if USE procedures are present in the module, and compiled code places its address in COB\$A_SF_USE. It is used by COB\$IOEXCEPTION to search for an applicable USE procedure if an I/O exception occurs.

MACRO

COB\$A_USE_PNC= 0,0,32,0 %,	! Address of PERFORM nest counter
COB\$A_USE_MODES=4,0,0,0 %,	! Base of OPEN mode entries, in order:
	! INPUT
	! OUTPUT
	! I-O
	! EXTEND
COB\$B_USE_COUNT=36,0,8,0 %,	! (Same order as COB\$B_CTX_MODE)
COB\$B_GUSE_COUNT=36,8,8,0 %,	! Count of file entries
COB\$A_USE_FILES=40,0,0,0 %:	! Count of global file entries
	! Base of first file entry

!+ Definition for one USE description entry within the structure above.
 ! Mode entries begin at offset COB\$A_USE_MODES and are two longwords each.
 ! File entries begin at offset COB\$A_USE_FILES and are three longwords each.

MACRO

COB\$A_USE_PROC= 0,0,32,0 %,	! Address of procedure
COB\$A_USE_EOPR= 4,0,32,0 %,	! Address of end of PERFORM range block
COB\$A_USE_RAB= 8,0,32,0 %:	! Address of RAB (only in file entries)

LITERAL

COB\$\$_USE_MODES=8,	! Size of an open mode entry
COB\$\$_USE_FILES=12:	! Size of a file entry

!+ Definition of offsets of USE procedure addresses in signal arguments.

MACRO

COB\$A_FIL_PROC = 12,0,32,0 %,	! Addr of file specific USE procedure
COB\$A_OPN_PROC = 16,0,32,0 %:	! Addr of open mode specific USE procedure

!+ Definition for the COBOL specific stack location (relative to FP, mapped as
! a REF BLOCK[.BYTE]).

MACRO

COB\$A_DB_USE = -8,0,32,0%;

! \Accesses offset 8 from the stack
! which contains a ptr to the
! /Data Base USE list

+ Definition for the Data Base USE description list pointed to by COB\$A_DB_USE.
 This structure is allocated by the compiler in PSECT \$PDATA if USE procedures are
 present in the module, and compiled code places its address in COB\$A_DB_USE.
 It is used by COB\$DBEXCEPTION to search for an applicable USE procedure if
 a Data Base exception occurs.

MACRO

COB\$B_USE_CODE = 0,0,8,0%	! Identifying code for data base exceptions
COB\$A_DBUSE_PNC = 4,0,32,0%	! Address of PNC for declaring program
COB\$B_DBUSE_CNT = 8,0,8,0%	! # of DB USE procedures defined ! in this program (includes both local ! and global procedures defined in both ! this program and in containing programs
COB\$B_GDBUSE_CNT = 8,8,8,0%	! # of global DB USE procedures ! defined in the local program
COB\$A_DBUSE_ENT = 12,0,32,0%	! Base address of 1st DB USE procedure

+ Definition for one Data Base USE description entry within the
 structure above. Note that COB\$A_USE_PROC and COB\$A_USE_EOPR
 are already defined in the USE list for routine COB\$IOEXCEPTION.
 Also, note that COB\$B_USE_LIT occupies the same field reference as
 COB\$A_USE_RAB does for routine COB\$IOEXCEPTION.

MACRO

COB\$A_USE_PROC = 0,0,32,0 %	! Address of procedure
COB\$A_USE_EOPR = 4,0,32,0 %	! Address of end of PERFORM range block
COB\$B_USE_LIT = 8,0,32,0 %	! A data base exception literal or zero ! for 'ON OTHER' or no 'ON'

LITERAL

COB\$K_DBUSE_CODE = 1,	! Code indicating generic class ! = Data Base
COB\$S_DBUSE = 12;	! Size of a DB entry

+ Definition of offset of Data Base USE procedure address in the
 signal argument vector. (Used in COB\$HANDLER when trying to find an
 applicable USE procedure.

MACRO

COB\$A_DBCHK_PROC = 12,0,32,0%;

!+
! Definitions for the first parameter of COBSIOEXCEPTION, which encodes the
! statement type and the specific error that has been encountered. Note
! that if the error-type parameter indicates an RMS error, the RMS status is
! obtained from the RMS structures.
!-

MACRO

COBSW_EXC_STMT= 0,0,16,0 %; ! I/O statement type
COBSW_EXC_ERROR=2,0,16,0 %; ! Error type

!+
! Values for COBSW_EXC_STMT.
!-

LITERAL

COBSK_EXC_MINS=1, ! Minimum for CASEing

!+
! Class of open errors, varying in file organization
! and file access.
!-

COBSK_EXC_OPESS= 1,

COBSK_EXC_OPERS= 3,

COBSK_EXC_OPERR= 4,

COBSK_EXC_OPEIS= 5,

COBSK_EXC_OPEIR= 6,

!+
! Class of close errors, varying in file organization
! and file access.
!-

COBSK_EXC_CLOSS= 7,

COBSK_EXC_CLORS= 9,

COBSK_EXC_CLORR=10,

COBSK_EXC_CLOIS=11,

COBSK_EXC_CLOIR=12,

!+
! Class of read errors, varying in file organization
! and file access.
!-

COBSK_EXC_REASS=13,

COBSK_EXC_REARS=15,

COBSK_EXC_REARR=16,

COBSK_EXC_REAIS=17,

COBSK_EXC_REAIR=18,

!+
! Class of write errors, varying in file organization

! and file access.
!-

COBSK_EXC_WRISS=19,

COBSK_EXC_WRIRS=21,

COBSK_EXC_WRIIR=22,

COBSK_EXC_WRIIS=23,

COBSK_EXC_WRIIR=24,

!+
! Class of re-write errors, varying in file organization
! and file access.
!-

COBSK_EXC_REWSS=25,

COBSK_EXC_REWRS=27,

COBSK_EXC_REWRR=28,

COBSK_EXC_REWIS=29,

COBSK_EXC_REWIR=30,

!+
! Class of delete errors, varying in file organization
! and file access.
!-

COBSK_EXC_DELSS=31,

COBSK_EXC_DELRS=33,

COBSK_EXC_DELRR=34,

COBSK_EXC_DELIS=35,

COBSK_EXC_DELIR=36,

!+
! Class of start errors, varying in file organization
! and file access.
!-

COBSK_EXC_STARS=39,

COBSK_EXC_STAIS=41,

!+
! Close re-wind error in sequential organization and
! sequential access.
!-

COBSK_EXC_CLRSS=43,

COBSK_EXC_UNLCK=44,

COBSK_EXC_MAXS=44 ;

! Maximum for CASEing

!+
! Values for COBSW_EXC_ERROR.

!-
LITERAL

COBSK_EXC_MINM= 0,
COBSK_EXC_RAB= 0,
COBSK_EXC_FAB= 1,
COBSK_EXC_FLK= 2,
COBSK_EXC_OPN= 3,
COBSK_EXC_MIN= 4,

COBSK_EXC_ORG= 5,

COBSK_EXC_KEY = 6,

COBSK_EXC_PIO = 7,

COBSK_EXC_MAXM= 7;

! Minimum for CASEing
! RMS RAB error.
! RMS FAB error.
! OPEN of a file closed WITH LOCK.
! OPEN of a file that is already open.
! The length of a variable length record
! is less than the minimum specified.
! This occurs on READs, WRITEs, and REWRITEs.
! The organization of the file opened
! does not match what was expected.
! An index key of the file opened does
! not match what was expected
! Previous IO operation was NOT successful READ
! Maximum for CASEing

!+ Definitions for the first parameter used by the COB\$ACCEPT and COB\$DISPLAY
run-time routines. These routines are invoked by the in-line code compiled
for the ACCEPT and DISPLAY statements, respectively. This parameter is the
integer unit encodings designating the unit from which input/output is to read/written
for these statements.
!-

LITERAL

COB\$K_UNIT_MIN= 0,	! Minimum for CASEing
COB\$K_INPUT= 0,	! COB\$INPUT logical name
COB\$K_OUTPUT= 1,	! COB\$OUTPUT logical name
COB\$K_CONSOLE= 2,	! COB\$CONSOLE logical name
COB\$K_CARDREAD= 3,	! COB\$CARDREADER logical name
COB\$K_PTREADER= 4,	! COB\$PAPERTAPERREADER logical name
COB\$K_LINEPRINT= 5,	! COB\$LINEPRINTER logical name
COB\$K_PTPUNCH= 6,	! COB\$PAPERTAPEPUNCH logical name
COB\$K_UNIT_MAX= 6;	! Maximum for CASEing

!+ Buffer size allocated in COB\$ACCEPT. Represents largest record that
can be ACCEPTed.
!-

LITERAL

COB\$K_ACC_SIZE = 1024 ;

!+ Buffer size of temp buffer in COB\$DISPLAY with which it will concatenate
callers strings. Beyond this size, it resorts to heap storage.
!-

LITERAL

COB\$K_DIS_SIZE = 132 ;

0060 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TYPEMAIN
LIS

COBPROLOG
REQ

COBACCDAT
LIS

COBACCDWK
LIS

UNLOCK
LIS

UTILSUBS
LIS

INTPAR
SOL

COBDEF
REQ

COBACCDAY
LIS

COBACCECU
LIS

COBRTL

COBLNK
REQ

COBRTL
MAP